ARTIC WOLF: SAVING THE FAMILY

David Agudelo Ochoa

Erika Dayana León Quiroga

Universidad de Antioquia
Despartamento de Ingeniería Electrónica y Telecomunicaciones
Informática II
Medellín-Antioquia
Abril de 2021

Índice

1. Sección introductoria.

En este informe veremos el desarrollo del proyecto final de la materia Informática II, el cual será un juego llamado Gartic Wolf: Saving the family. Dentro de las secciones veremos listas de tareas hechas para organizar mejor el trabajo a realizar, actualizaciones y problemas del código del programa.

2. Lista de tareas.

- Jueves 7 de Abril:
  - Crear proyecto Qt
  - Crear la clase ObjetoAnimado y las que heredarán de ella.
  - Objeto animado: Atributos y métodos.
  - Método StringPath.
  - Probar método animar.
  - Instalar librería QMediaPlayer (para sonidos del juego.)
- Viernes 8 de Abril:
  - Agregar atributos y constructor de las clases que heredan de Objeto animado, como héroe, enemigo, proyectil, reloj.
  - Creación y animación del escenario y el héroe.
  - Pruebas de la animación.
- Sábado 9 de Abril:
  - Agregar métodos de las clases que heredan de Objeto animado, como héroe, enemigo, proyectil, reloj.
  - Método de proyectil que mueve y verifica la colisión con otros objetos del mismo.
  - Agregar el método que creará el nivel uno.
- Domingo 10 de Abril:
  - Buscar ecuaciones de péndulo.
  - Conectar las vidas, y el tiempo con la finalización del nivel.
  - Agregar el menú.
  - Personalizar el menú.
- Lunes 11 de Abril:
  - Modelar el comportamiento del péndulo.
  - Terminar nivel uno.
- Martes 12 de Abril:
  - Empezar a trabajar en el guardado del juego.
  - Ecuaciones para el resorte y el movimiento parabólico.
- Miércoles 13 de Abril:
  - Modelar proyectil de nivel dos.
- Jueves 14 de Abril:
  - Modelar el comportamiento del péndulo.
  - Cargar nivel 2.

3. Desarrollo del proyecto.

Para el desarrollo del proyecto Artic Wolf: Saving the Family, se tuvo en cuenta el informe de Clases que se realizó previamente con la intención de tener un plan de organización de las clases que serían necesarias para el funcionamiento del juego que se tiene planeado. En este informe de clases encontramos la descripción nivel por nivel del juego y la descripción clase por clase del mismo. A continuación veremos los cambios que se le realizaron a las clases que se tenían pensadas a la hora de materializar estas ideas en código.

En los días **Jueves 7 y Viernes 8 de abril** se comenzó por crear el proyecto con la clase padre Objeto animado con sus respectivos atributos y métodos, y sus clases hijas Heroe, Enemigo, Proyectil y Reloj. En la clase objeto animado se realizaron varios cambios con respecto a lo planeado en el informe de clases. Esto porque a medida que se va escribiendo el código se encuentran formas más efectivas de realizar ciertas tareas que las que ya se tenían planeadas. Sin embargo, el uso del informe de Clases es fundamental para la asignación de tareas y la realización del avance del proyecto.

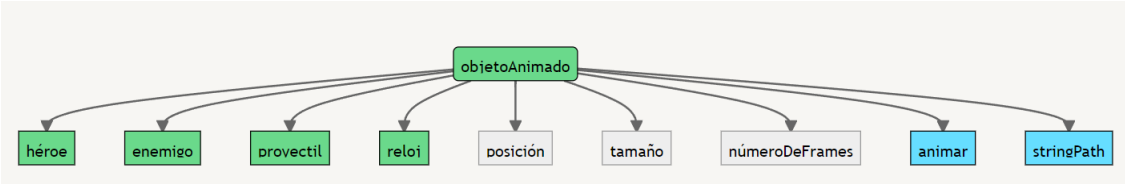

Figura 1: Clase ObjetoAnimado que se tenía pensada construir.

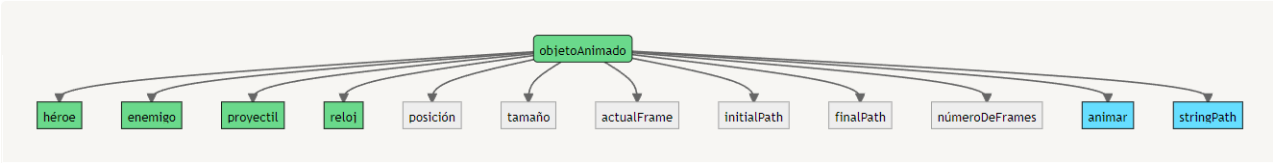

Figura 2: Clase ObjetoAnimado construida.

Con los cambios realizados en la clase objeto animado se logró obtener el movimiento esperado en los objetos que se muestran en la escena, esto con ayuda del método animar y stringPath los cuales siguen cumpliendo las mismas funciones que se les asignaron en la parte de planeación del proyecto. El método stringPath recibe el número de la imagen de la cuál se requiere su path y luego lo retorna. La función animar utiliza el método anterior para conocer el camino a la imagen y haciendo uso de actualFrame como parámetro del método stringPath logra ir cambiando el frame de tal forma que cuando sea llamada, estos varíen y den la impresión de movimiento.

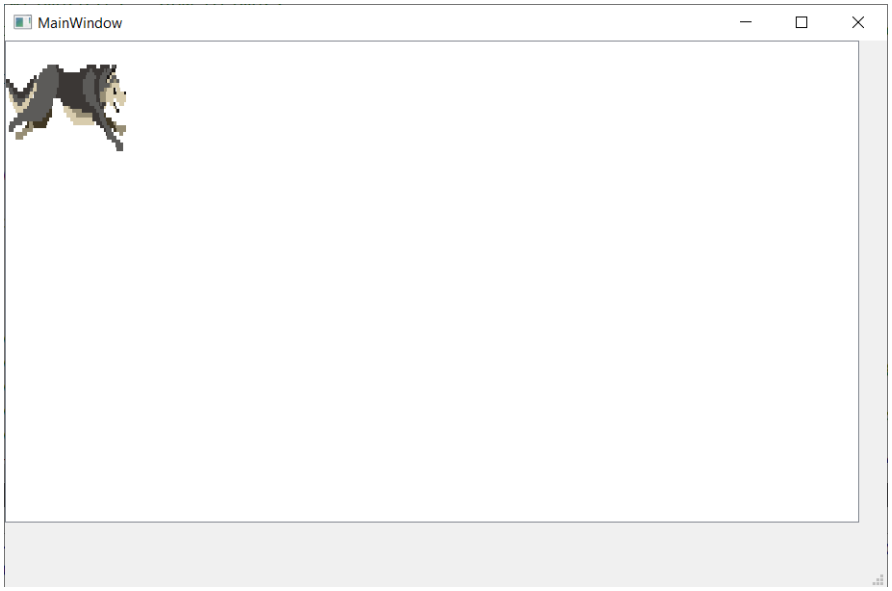

Figura 3: Pruebas del funcionamiento de la clase objeto animado.

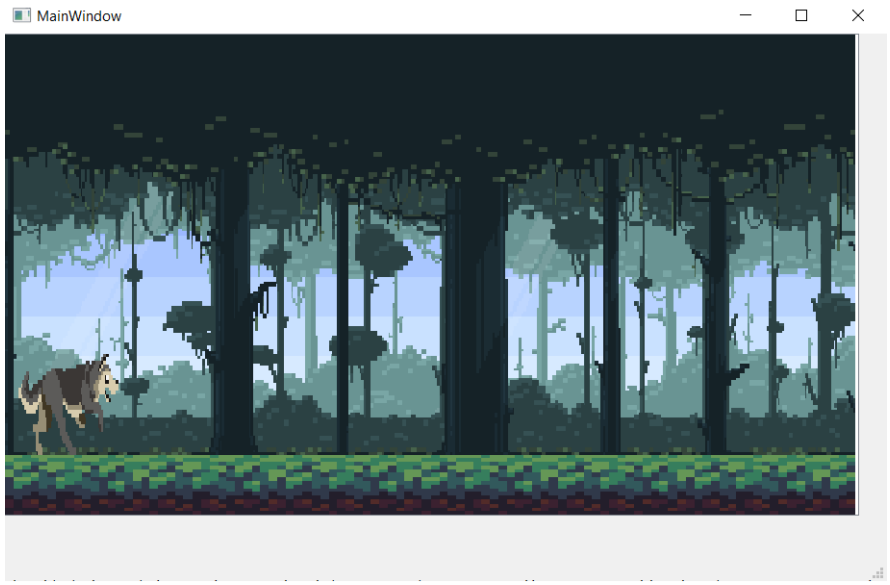

Figura 4: Pantalla de prueba con el héroe y el fondo animados.

El **Sábado 9 de abril** se empezó con la creación de la clase padre Nivel y sus clases hijas nivel1 y nivel2 como se planeó en el informe de Clases. Estas clases se utilizarían para armar el nivel correspondiente en cada clase.

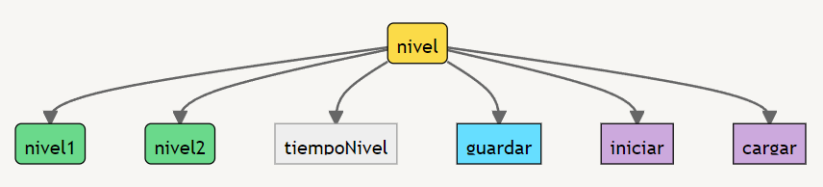

Figura 5: Clase nivel planeada en el informe de clases.

Sin embargo, nos dimos cuenta que la función que le otorgamos a estas clases en la planeación podría ser reemplazada por un **método en el MainWindow** que realice la misma función, en este método se agregarían todos los objetos necesarios para crear el nivel correspondiente. Por lo tanto la clase padre nivel y sus hijas son descartadas.

Este mismo día se comienza con el modelamiento del proyectil del nivel 1, para esto utilizamos el planteamiento de la clase proyectil que se realizó en el informe de clases.

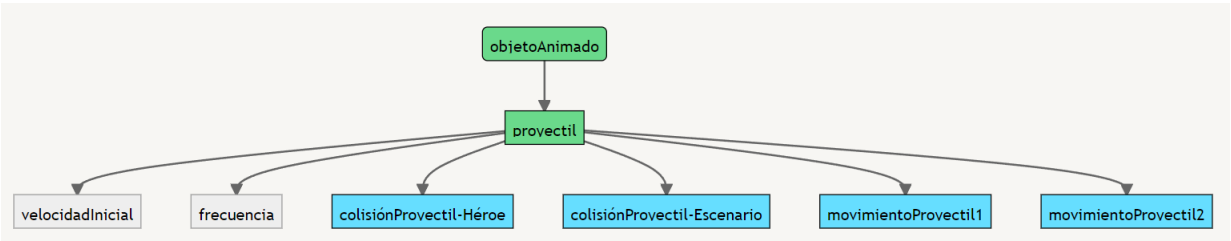

Figura 6: Clase nivel planeada en el informe de clases.

Se trabaja en la colisión entre el proyectil y el escenario, tratando de que cada que el proyectil salga de este se borre de la escena y se cree un nuevo proyectil, pero el borrado de este objeto provocaba conflictos en el proyecto por lo que se decidió que no se borraría el objeto sino que se le haría una especie de Reset a su posición, de esta forma no se estarían creando y borrando nuevos objetos sino que siempre es el mismo, y cuando sale de la escena o se choca con el héroe vuelve a su posición inicial. El movimiento automático del proyectil se logra haciendo uso de un Timer y conectándolo con las ecuaciones de movimiento del proyectil del nivel 1.

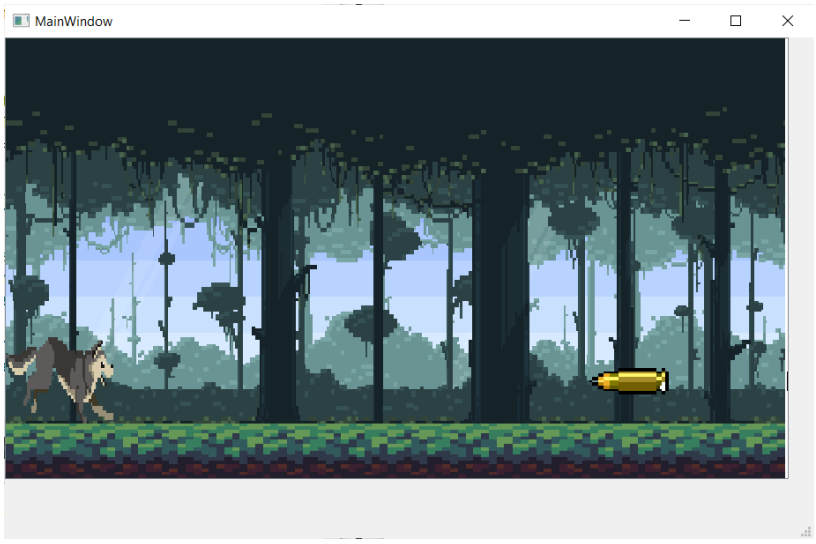

Figura 7: Pruebas de la animación y la colisión entre el proyectil y la pared.

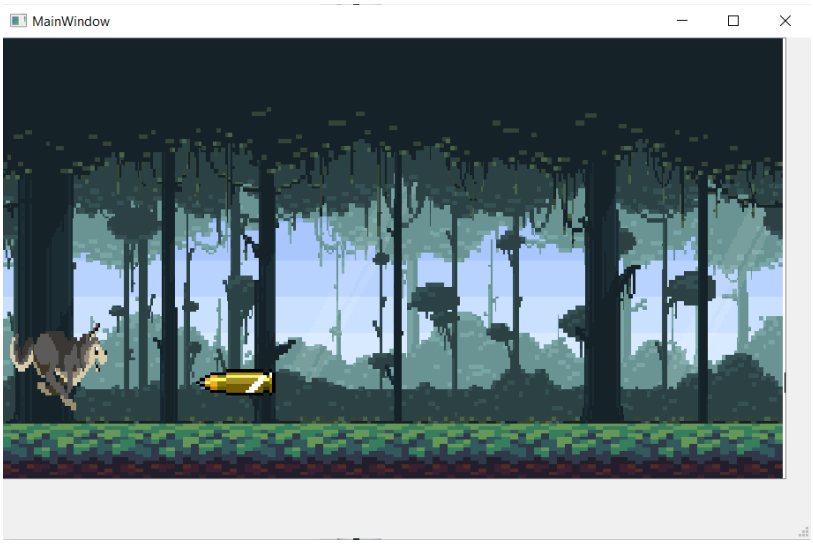

Figura 8: Pruebas de la animación y la colisión entre el proyectil y la pared.

El **Domingo 10 de abril** se agregó la animación del enemigo, se ajustó la ventana de Graphics-View y el tamaño de todos los objetos en la escena.

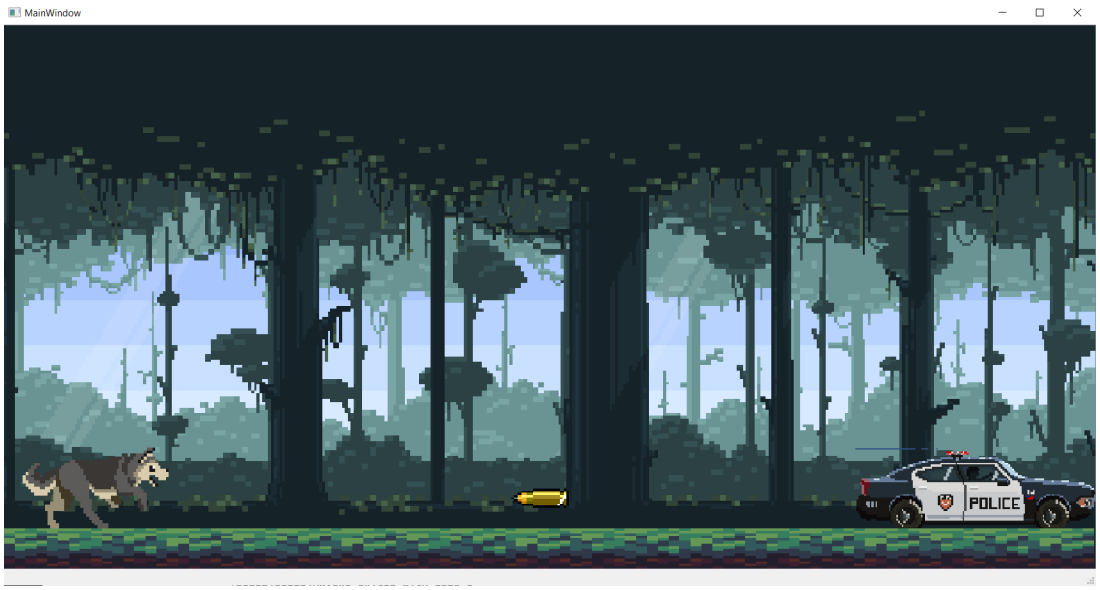

Figura 9: Ajuste del tamaño de los objetos de la escena.

También se agregó el contador de las vidas y del tiempo de la partida como podremos ver en la siguiente imagen (11), estos contadores se ubicarán en la parte superior izquierda de la escena, en la imagen de demostración están encerrados en un recuadro rojo para ubicarlos de la manera más sencilla. El tiempo irá disminuyendo conforme transcurre la partida con ayuda de un Timer. Las vidas disminuirán cada que un proyectil choque con el héroe y esto será codificado en el método de la clase proyectil en donde se verifica la colisión entre el héroe y el proyectil.

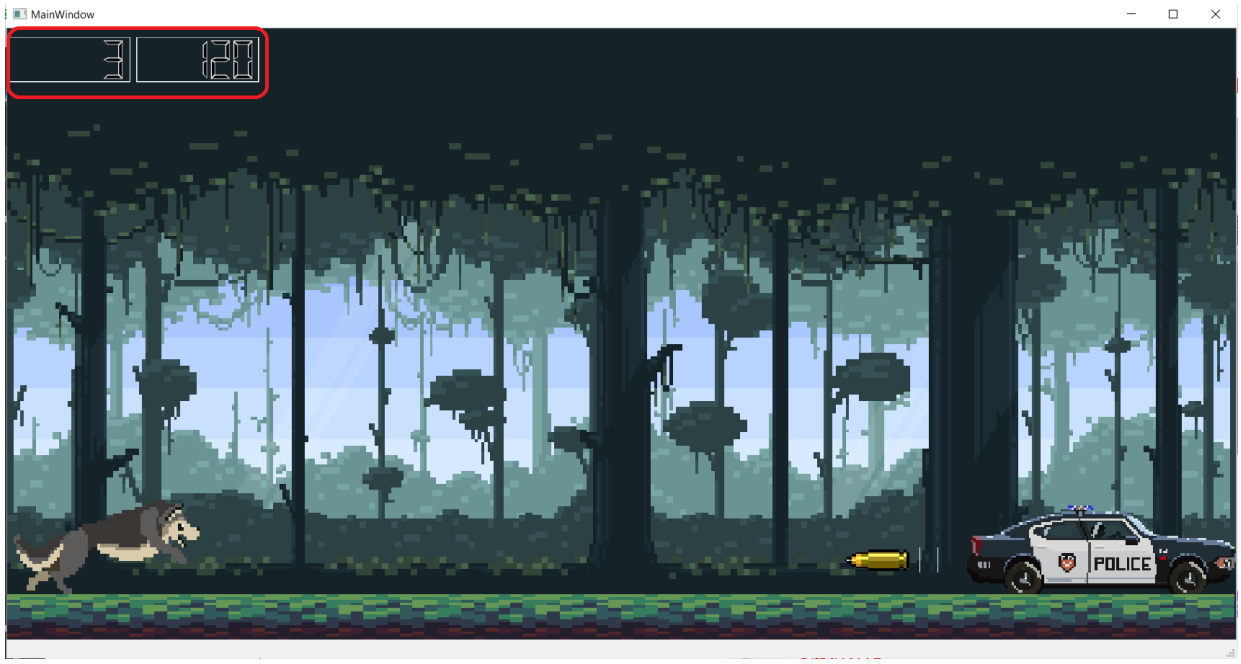

Figura 10: Reloj y vidas del héroe puestas en pantalla.

Se hicieron pruebas para agregar la pantalla del menú, la cuál aparecerá cuando se corra el juego y una vez se presione la techa I del teclado iniciará la partida con tres vidas y 120 segundos en el contador del tiempo de la partida. Esto se hará con un KeyPressEvent y un if que al presionar la tecla determinada iniciará el método del MainWindow encargado de crear los objetos del nivel 1.

```
if(i->key() == Qt::Key_I){
    cargarNivel1();
}
```

Figura 11: Condicional encargado de iniciar el nivel 1 cuando se presione la tecla I.

4. Conclusiones.

-